*PATENT*

UNITED STATES PATENT APPLICATION

FOR

# METHOD AND APPARATUS FOR SCALABLE DISAMBIGUATED COHERENCE IN SHARED STORAGE HIERARCHIES


INVENTORS:

SUJAT JAMIL
HANG NGUYEN
QUINN MERRELL

PREPARED BY:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1026

(408) 720-8598


Attorney Docket No. 42P8931C

**EXPRESS MAIL CERTIFICATE OF MAILING**

"Express Mail" mailing label number EV 341 064 508 US

Date of Deposit July 15, 2003

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to Mail Stop Patent Application, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

Beverly Kehoe Shea
(Typed or printed name of person mailing paper or fee)

Beverly Kehoe Shea 7/15/03
(Signature of person mailing paper or fee) Date

# METHOD AND APPARATUS FOR SCALABLE DISAMBIGUATED COHERENCE IN SHARED STORAGE HIERARCHIES

## CROSS-REFERENCE TO RELATED APPLICATIONS

**[0001]** This application is a continuation and claims the benefit of application Ser. No. 09/677,122, filed September 29, 2000, currently pending.

## FIELD OF THE INVENTION

**[0002]** This invention relates generally to shared storage hierarchies in multiprocessing systems, and in particular to use of an exclusive dirty status in a coherence protocol to disambiguate ownership and modification status for memory references in a shared multi-level storage hierarchy.

## BACKGROUND OF THE INVENTION

**[0003]** In a multiprocessing system with a shared multi-level storage hierarchy, typically comprising a shared cache storage, one processor may request access from a shared cache storage to data that is in a state of ownership by another processor. The requesting processor does not know if the requested data in the shared cache storage is valid or if it has been modified by another processor in a private storage at another level of the storage hierarchy. Therefore the requested data in the shared cache storage is not useful to the requesting processor until its actual status can be

determined. An ambiguous status for data in shared cache storage hierarchies is referred to as a "sharing ambiguity."

**[0004]** One method for resolving sharing ambiguities in multiprocessing systems makes use of a common bus to snoop transactions (M. Papamarcos and J. Patel, "A Low-Overhead Coherence Solution for Multiprocessors with Private Cache Memories," Proc. 11th ISCA, 1984 pp. 348-354). Snooping transactions on a common bus to maintain coherence increases traffic to a processor's private cache storage. This increased traffic is not necessarily related to data that is actually needed by the respective processor. Therefore a disadvantage of snooping is that the average latency of cache accesses is increased since requests for data have to compete with snoops for access to the private caches. Moreover, the competition increases with the number of processors sharing a common bus. As a consequence, overall system performance suffers due to slower average access times. A second disadvantage of snooping can occur because an access to requested data in the shared cache storage must wait until results of snooping are received.

**[0005]** Another method for resolving sharing ambiguities involves broadcasts of inquiries (commonly referred to as disambiguating inquiries or backward inquiries) over an interconnection network shared by the processors with access to the shared cache storage. When requested data is found to be in an ambiguous state, an inquiry is broadcast to the other processors. Again, latency increases since the requesting processor must wait until responses to the broadcast inquiry are received. As the number of processors sharing a

cache storage increases, so does the potential number of broadcasts and responses—contributing to increased network congestion.

**[0006]** In addition to the increases in latency associated with these prior methods, there are also additional costs associated with providing hardware functionality in order to implement a particular chosen method. Hardware functionality requires additional circuitry, and additional circuitry requires increased silicon area. An undesirable secondary effect of additional hardware circuitry and increased silicon area is an increase in the number and severity of critical timing paths, potentially resulting in further performance degradation for the overall system.

**[0007]** Another method used in distributed systems is known as SCI (Scalable Coherent Interface, IEEE Std 1596-1992 *Scalable Coherent Interface*, Piscataway, NJ). SCI supports a one-writer-multiple-reader format with a distributed doubly linked list that is maintained through main memory. Addresses of private cache storage are inserted onto the list in a controlled manner and only the address at the head of the list may overwrite the data. The interface maintains a coherent storage hierarchy, by forwarding data requests to the head of the list. One disadvantage of such a system is that before data may be overwritten the list must be sequentially purged. For large distributed systems, delays associated with such a method potentially contribute to performance degradation of the overall system. In addition to the potential for undesirable network congestion inherent in such a distributed

system, problematic issues of link maintenance in cases of distributed system failures must also be addressed.

BRIEF DESCRIPTION OF THE DRAWINGS

**[0008]** The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings.

**[0009]** **Figure 1** shows a multiprocessing system with shared and private storage.

**[0010]** **Figure 2** illustrates an example of sharing ambiguities in a multiprocessing system with shared and private storage.

**[0011]** **Figure 3** shows a state transition diagram for one embodiment of a coherency protocol for resolving sharing ambiguities, the protocol including an exclusive-dirty status and an exclusive-clean status.

**[0012]** **Figure 4** illustrates one embodiment of a coherent storage hierarchy including private storage and shared storage that supports resolving sharing ambiguities without backward inquiries.

**[0013]** **Figure 5** illustrates one embodiment of a method for resolving sharing ambiguities in accordance with the coherent storage hierarchy of Figure 4.

**[0014]** **Figure 6** illustrates an example of a multiprocessing system using the coherent storage hierarchy of Figure 4.

**[0015]** **Figure 7** illustrates another embodiment of a coherent storage hierarchy including private storage and shared storage that supports resolving sharing ambiguities.

**[0016]** **Figure 8** illustrates one embodiment of a method for resolving sharing ambiguities in accordance with the coherent storage hierarchy of Figure 7.

**[0017]** **Figure 9a** illustrates an alternative embodiment of a coherent storage hierarchy including private storage and shared storage that supports resolving sharing ambiguities.

**[0018]** **Figure 9b** illustrates another alternative embodiment of a coherent storage hierarchy including private storage and shared storage that supports resolving sharing ambiguities.

**[0019]** **Figure 9c** illustrates another alternative embodiment of a coherent storage hierarchy including private storage and shared storage that supports resolving sharing ambiguities.

**[0020]** **Figure 9d** *illustrates another alternative embodiment of a coherent* storage hierarchy in a networked distributed system including private storage and shared storage that supports resolving sharing ambiguities.

**[0021]** **Figure 10** illustrates an embodiment of a computing system including a coherent storage hierarchy comprising private storage using a first coherence protocol, shared storage using a second coherence protocol that supports resolving sharing ambiguities and shared storage using bus snooping and a third coherence protocol.

**[0022]** **Figure 11** illustrates another embodiment of a computing system including a coherent storage hierarchy comprising private storage using a first coherence protocol, shared storage using a second coherence protocol that

supports resolving sharing ambiguities and distributed shared storage using a third coherence protocol.

**[0023]** **Figure 12** illustrates another embodiment of a computing system including a coherent storage hierarchy comprising private storage using a distributed coherence protocol, distributed shared storage also using a distributed coherence protocol, and distributed shared storage using a second coherence protocol that supports exclusive locally isolated ownership while resolving sharing ambiguities.

**[0024]** **Figure 13a** illustrates an embodiment of a multiple-core processor including a coherent storage hierarchy comprising private storage and shared storage that supports resolving sharing ambiguities.

**[0025]** **Figure 13b** illustrates another embodiment of a computing system including a coherent storage hierarchy comprising private storage and shared storage that supports resolving sharing ambiguities.

DETAILED DESCRIPTION

**[0026]** For one embodiment of a coherent shared storage hierarchy in a multiprocessor system, a shared data storage unit provides storage portions to hold data, which may be called data portions, and storage portions to hold corresponding status encodings, which may be called status portions. The shared storage unit may be, for example, a shared cache or a shared file storage or a shared data base or some other shared storage, and the data portions may be cache lines or files or records or some other portion of data respectively. As it is common to refer to a cache line and the data stored in a cache line synonymously, or to refer to a file and the data stored in a file synonymously; a data portion and the data stored in a data portion may be used interchangeably if, for example, no distinction is being made between the two.

**[0027]** A status encoding corresponding to a particular data portion provides enough information to disambiguate a data request to shared storage without resorting to prior methods of snooping a shared bus or of transmitting backward status inquiries to private storage.

**[0028]** Shared-storage control transmits the data from shared storage in response to the data request if its corresponding status encoding indicates a clean status, meaning that no private copies of the requested data have been modified. On the other hand, shared-storage control transmits a data request to a private storage unit if the corresponding status encoding indicates an exclusive dirty status, meaning that a copy of the requested data in private

storage has been modified. The private storage unit, in turn, provides a coherent copy of the requested data to the shared storage unit. Shared-storage control then proceeds to satisfy the initial data request with the coherent copy provided by the private storage unit. Alternatively, the coherent copy could be provided directly by the private storage unit to the shared storage unit and to satisfy the initial data request simultaneously or in either order.

**[0029]** If a data request to the shared storage unit indicates a need to modify the requested data, shared-storage control also provides for invalidation transmissions to private storage units that have previously requested copies of the relevant data. The status encoding corresponding to a portion comprising such data is then set to an exclusive dirty status to provide means for coherently processing potential future data requests.

**[0030]** These and other embodiments of the present invention may be realized in accordance with the following teachings and it should be evident that various modifications and changes may be made in the following teachings without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than restrictive sense and the invention measured only in terms of the claims.

**[0031]** Figure 1 shows a multiprocessing system 100 with shared storage 190, which provides data to private storage 110 for processor 101 and to private storage 120 for processor 102. Multiprocessing system 100 may be on

a single or on multiple printed circuit boards. Alternatively, multiprocessing system 100 may be on a multiple-processor-core silicon die, or in a multi-chip module, or distributed on a communication network.

**[0032]** A coherent storage hierarchy is one in which a processor request to read a portion of data that has previously been modified somewhere in the hierarchy results in an up-to-date copy of the modified data portion being supplied to the requesting processor. Coherence protocols are used to guarantee such behavior in a storage hierarchy. One such coherence protocol uses four states: M (modified), E (exclusive), S (shared) and I(invalid). A detailed description of the MESI protocol may be found in (Lilja, D. J.. Cache Coherence in Large-Scale Shared-Memory Multiprocessors: Issues and Comparisons, ACM Computing Surveys, 25(3), September 1993).

**[0033]** As described above, sharing ambiguities in such systems must be disambiguated through some means before a processor request may be satisfied in a coherent manner.

**[0034]** Figure 2 illustrates an example of sharing ambiguities in a multiprocessing system 200 with shared storage 290 and private storage 210 and 220. For instance, if data portion 219 has a status of M in shared storage 290 and a processor 202 has previously requested a data portion copy 218, it is ambiguous as to which copy (data portion 219 or data portion 218) should be provided in response to a subsequent data request for data portion 219 from processor 201. If processor 202 has not modified data portion 218 then shared storage 290 may provide a copy of data portion 219 to the private storage 210

for processor 201. This would provide faster access to data portion 219 and contribute to better overall system performance. But if processor 202 has modified data portion 218, then a data request should be sent from shared storage 290, to private storage 220 for an updated copy of data portion 218, with which to satisfy the data request of processor 201. The modified status of data portion 219 is ambiguous. In order for shared storage 290 to respond to the data request, the status of data portion 218 in private storage 220 needs to be known to shared storage 290.

**[0035]** A similar situation exists for data having an exclusive status, E. For instance, if data portion 217 has a status of E in shared storage 290 and processor 201 has previously requested a data portion copy 216, it is ambiguous as to which copy (data portion 217 or data portion 216) should be provided in response to a subsequent data request for data portion 217 from processor 202. If processor 201 has not modified data portion 216 then shared storage 290 may provide a copy of data portion 217 to the private storage 220 for processor 202. Again, this would provide faster access to data portion 217 and contribute to better overall system performance. But if processor 201 has modified data portion 216, then a data request should be sent from shared storage 290, to private storage 210 for an updated copy of data portion 216, with which to satisfy the data request of processor 202. The exclusive status of data portion 217 is ambiguous. In order for shared storage 290 to respond to the data request, the status of data portion 216 in private storage 220 needs to be known to shared storage 290.

[0036] It will be appreciated that the M status and the E status are both exclusive statuses when a processor has a right to modify the data, since both indicate sole ownership. The difference is that the M status indicates a need to update the central storage or main memory. In order to resolve sharing ambiguities and avoid backward inquiries the status of a data portion is tracked in the shared storage. What is needed is a way to indicate that these exclusive statuses require data updates from peripheral private storage.

[0037] Figure 3 shows a state transition diagram 300 for one embodiment of a coherency protocol for resolving sharing ambiguities, the protocol including an exclusive-dirty status (ED) and an exclusive-clean status (EC). Prior to a data request a data portion may be given a default status of I, or in response to an invalidation request a data portion may be returned to a status of I by transition 314 from a status of M, transition 324 from a status of ED, transition 334 from a status of EC, or transition 354 from a status of S. Following a data request to a central storage or a main memory, a data portion may be supplied to shared storage having a status of EC according to transition 343, or having a status of S according to transition 345. If the data request is to modify the data portion a status of ED may be assigned according to transition 342.

[0038] Following a data request from a private storage, a data portion having an initial status of EC may retain a status of EC according to transition 333 if no data portion copies reside in a private storage other than the requesting private storage. It may be assigned a new status of S according to transition 335 if a data portion copy resides in a private storage other than the

requesting private storage, or it may be assigned a new status of ED if the data request from private storage is a request for ownership with a right to modify the data portion. A data portion having an initial status of S may be reassigned a status of S according to transition 355 if the data request from private storage is not a request for ownership with a right to modify the data portion, or it may be assigned a new status of ED according to transition 352 if the data request is a request for ownership with a right to modify. It will be appreciated that a transmission of invalidation requests to private storage may also be associated with transition 352.

**[0039]** If a data portion is written back to shared storage from private storage, an initial status of ED may be assigned a new status of M according to transition 321, and conversely a data portion having an initial status of M may be assigned a new status of ED following a data request for ownership with a right to modify the data portion. A data portion having an initial status of ED may retain or return to a status of ED according to transition 322 following a data request for ownership with a right to modify the data portion. It will be appreciated that other associated activities, such as requesting a updated copy of the data portion, and transmitting invalidation requests may accompany transition 322 and that interim temporary status assignments may be used in conjunction with these associated activities. Finally, a data portion having an initial status of M may retain a status of M according to transition 311 following a data request that is not a request for ownership with a right to modify the data

portion until the data portion has been written back to central storage or main memory.

**[0040]** It will be appreciated that a set of five statuses may be encoded using three or more bits. It will also be appreciated that a set of dirty status encodings could also include a modified-dirty (MD) status encoding to indicate a need to receive a data update from a peripheral private storage having ownership with a right to modify, and that a set of statuses including a sixth status could conveniently be encoded without requiring additional encoding bits. It will further be appreciated that some implementations may find it beneficial to also include a sixth status.

**[0041]** For one embodiment of a coherence protocol for disambiguated coherence in shared storage hierarchies, a method is herein disclosed that provides for indicating one or more exclusive dirty statuses for a data portion in a shared storage to track ownership in a private storage thereby disambiguating the shared status, reducing backward inquiries and improving overall system performance.

**[0042]** It will be appreciated that a further reduction in backward inquiries may be garnished if the identity of a private storage owner of a data portion having an exclusive dirty status was also known to the shared storage. For one embodiment of a coherent storage hierarchy, Figure 4 illustrates a scalable presence encoding to be held in a presence portion of storage for a corresponding data portion that further reduces backward inquiries and potentially avoids unnecessary invalidation requests.

[0043] Shared storage 490 stores data portions 411 through 430. Each data portion has, in storage control 491, a corresponding status portion to hold a status encoding and a corresponding presence portion to hold a presence encoding. When a data request is received from private storage 410, private storage 420, private storage 440, or private storage 480, a copy of the requested data portion is retrieved from a central storage or main memory and stored by shared storage 490. A corresponding status portion holds a status encoding of the data portion. A corresponding presence portion holds a presence encoding according to which private storage units have requested a copy of the data portion.

[0044] For example a status portion corresponding to data portion 411 holds a status encoding of ED and a corresponding presence portion holds a presence encoding of 0001 indicating that data portion 411 has been provided to private storage 410 in response to a data request of a type that indicates a need to modify the requested data portion. Similarly data portion 413 has a status encoding of ED and a presence encoding of 0100 indicating that data portion 413 has been provided to private storage 440 in response to a data request with a need to modify the data. Data portion 415 has a status encoding of ED and a presence encoding of 1000 indicating that data portion 415 has been provided to private storage 480 in response to a data request with a need to modify the requested data. And finally, data portion 418 has a status encoding of ED and a presence encoding of 0010 indicating that data

portion 418 has been provided to private storage 420 in response to a data request of a type that indicates a need to modify the requested data portion.

**[0045]** A status portion corresponding to data portion 412 holds a status encoding of S and a corresponding presence portion holds a presence encoding of 0101 indicating that data portion 412 has been provided to private storage 410 and to private storage 440 in response to data requests of a type that does not indicate a need to modify the requested data portion. Similarly data portion 419 has a status encoding of S and a presence encoding of 1111 indicating that data portion 419 has been provided to private storage 410, private storage 420, private storage 440 and private storage 480 in response to data requests of a type that does not indicate a need to modify the requested data portion.

**[0046]** Data portion 414 has a status encoding of M and the corresponding presence portion holds a presence encoding of 1000 indicating that the previously modified data portion 414 has been provided to private storage 480 in response to a data request that does not indicate a need to modify the data.

**[0047]** Data portion 416 has a status encoding of I and a presence encoding of 0000 indicating that data portion 416 is invalid in shared storage, in which case the data or any information about private storage which could have previously held a copy may or may not be accurate. If a notion of inclusion is enforced, then any private storage having a copy would have also invalidated it. Alternatively, the storage location could correspond to a data portion that was evicted from shared storage and subsequently the same storage location

was invalidated, in which case a copy of the data portion may reside in private storage.

**[0048]** Data portion 417 has a status encoding of EC and a presence encoding of 0010 indicating that data portion 417 has been provided exclusively to private storage 420 in response to a data request of a type that does not indicate a need to modify the requested data. Finally, data portion 430 has a status encoding of EC and a presence encoding of 0001 indicating that data portion 430 has been provided exclusively to private storage 410 in response to a data request that does not indicate a need to modify the requested data. It will be appreciated that copies of a data portion having a status encoding of EC may also be provided to multiple private storage units within the local storage hierarchy rooted at a particular shared storage.

**[0049]** The coherent storage hierarchy of Figure 4 provides for resolving sharing ambiguities without requiring a shared bus for snooping. It further provides for reducing backward inquiries and for reducing transmission of unnecessary invalidation requests, thereby contributing toward higher system performance.

**[0050]** Figure 5 illustrates one embodiment of a method for resolving sharing ambiguities in accordance with the coherent storage hierarchy of Figure 4. In processing block 500, processor i requests to read a data portion, the data request being received by processor i's private storage and processed by processing block 501. In processing block 501, if the data portion is valid in private storage, then control flow proceeds to processing block 510, which

provides the requested data portion to processor i. Otherwise, the data request is received by storage control 491 and processed by processing block 502. In processing block 502, if the status of the requested data portion is S, then control flow proceeds to processing block 509, which assigns a presence encoding to the data portion to indicate that a shared copy of the data portion will reside in the private storage of processor i. Otherwise, control flow proceeds to processing block 503. In processing block 503, if the status of the requested data portion is M, then control flow proceeds to processing block 509, which assigns a presence encoding to the data portion to indicate that a copy of the modified data portion will reside in the private storage of processor i. Otherwise, control flow proceeds to processing block 504. In processing block 504, if the status of the requested data portion is EC, then control flow proceeds to processing block 509, which assigns a presence encoding to the data portion to indicate that an exclusive copy of the data portion will reside in the private storage of processor i. Whenever control flow proceeds to processing block 509, the data portion in shared storage is an up-to-date copy and may be provided by shared storage in response to a data request.

**[0051]** If the status of the data portion is not EC then control flow proceeds to processing block 505. In processing block 505, if the status of the requested data portion is ED, then control flow proceeds to processing block 506, which requests an updated data portion from a private storage indicated by the corresponding presence encoding. Following the receipt of an updated copy, a new status encoding, of M for instance, may be reassigned to the data portion.

Alternatively the data portion may be written back to central storage or main memory and a new status of EC or S may be reassigned to the data portion. Control flow then proceeds to processing block 509, which assigns a presence encoding to the data portion to indicate that a copy of the data portion will reside in the private storage of processor i. Depending on the type of data request being processed, processing block 509 may also transmit invalidation requests to private storage previously indicated by the corresponding presence encoding. When processing has completed in processing block 509, control flow proceeds to processing block 510, which completes the transaction by supplying the requested data portion to the private storage of processor i.

**[0052]** If the status of the requested data portion is not ED in processing block 505, then control flow proceeds instead to processing block 507. In processing block 507, if the status of the requested data portion is I, then control flow proceeds to processing block 508, which requests an external data portion from central storage or main memory. Following the receipt of the requested data portion from central storage or main memory, control flow proceeds to processing block 509, which assigns a presence encoding to the data portion to indicate that a copy of the data portion will reside in the private storage of processor i. Control flow then proceeds to processing block 510, which supplies the requested data portion to the private storage of processor i.

**[0053]** It will be appreciated that the embodiment of Figure 5 is illustrated by way of example and not limitation and that the disclosed embodiment may be modified in arrangement and detail by those skilled in the art. For instance,

additional processing blocks may be included or combined to transmit invalidation requests, or to reassign new status encodings in accordance with a state transition diagram like the one shown in Figure 3. Additionally, the order with which control flow proceeds from processing block to processing block may be modified without departing from the principles of the disclosed embodiment within the scope of the accompanying claims. The methods and apparatus disclosed above may be employed in a multiprocessing system to provide a coherent storage hierarchy with reduced backward inquiries and reduced unnecessary invalidation requests.

**[0054]** Figure 6 illustrates an example of a multiprocessing system 600 using the coherent storage hierarchy of Figure 4, including processor 601 and its private storage 610, processor 602 and its private storage 620, processor 604 and its private storage 640, processor 608 and its private storage 680, and shared storage 690. Data portion 613, for example, has a status encoding of ED and a presence encoding of 0001 indicating that a modified data portion copy 612 resides in private storage 610 of processor 601. Data portion 625, for example, has a status encoding of EC and a presence encoding of 0010 indicating that an unmodified data portion copy 624 resides in private storage 620 of processor 602. If, for example, processor 608 requests a copy 686 of data portion 697 and copy 686 has a corresponding status of I then a request for data portion 697 is transmitted to shared storage 690. If the status of data portion 697 is ED and the corresponding presence encoding is 0100 indicating that a modified copy 646 resides in private storage 640, then a data request is

transmitted to private storage 640 for the modified copy 646. Private storage 640 may transmit the requested copy 646 and reassign it a status encoding of S. When the modified copy 646 is received by shared storage 690, data portion 697 may be updated and reassigned a status encoding of M. Updated data portion 697 may then be provided to private storage 680 and reassigned a presence encoding of 1100 to indicate that the two copies reside in private storage 640 and private storage 680. When data portion 697 is received by private storage 680, copy 686 may be updated and reassigned a status encoding of S. Thus the shared status of data portion 697 is disambiguated without snooping or unnecessary backward inquiries.

**[0055]** Figure 7 illustrates another embodiment of a coherent storage hierarchy that supports resolving sharing ambiguities. It will be appreciated that since shared storage 790 provides data portions to only two private storage units, private storage 710 and private storage 720, it is possible to disambiguate a data request without resorting to a presence encoding for each data portion being explicitly stored in control 791. If a particular private storage unit requests a data portion from shared storage, then an exclusive-dirty status encoding corresponding to that data portion indicates that the other private storage unit contains a modified copy of the requested data portion.

**[0056]** For example status portions corresponding to data portion 711, data portion 713, data portion 715 and data portion 718 hold status encodings of ED, so if the corresponding data portions are requested by one of the private storage units, then they would have already been provided to the other private

storage unit in response to a data request of a type that indicates a need to modify the requested data portion. Data portion 712 and data portion 719 have status encodings of S indicating that the corresponding data portion has been provided to one of the private storage units in response to a data request that does not need to modify the requested data portion. Data portion 714 has a status of M indicating that the previously modified data portion 714 may be provided in response to a data request that does not need to modify the requested data portion. Data portion 716 has a status of I indicating that data portion 716 has been invalidated in shared storage and any private storage which may have previously held a copy. Data portion 717 and data portion 730 have status encodings of EC indicating that the corresponding data portion has been provided in response to a data request of a type that does not need to modify the requested data portion.

**[0057]** Figure 8 illustrates one embodiment of a method for resolving sharing ambiguities in accordance with the coherent storage hierarchy of Figure 7. In processing block 800, processor i requests to read a data portion, the data request being received by the private storage of processor i and processed by processing block 801. In processing block 801, if the data portion is valid in private storage, then control flow proceeds to processing block 810, which provides the requested data portion to processor i. Otherwise, the data request is received by storage control 791 and processed by processing block 802. In processing block 802, if the data portion has a status of S, then control flow proceeds to processing block 810, which

completes the transaction by supplying the requested data portion to the private storage of processor i. Otherwise, control flow proceeds to processing block 803. In processing block 803, if the requested data portion has a status of M, then control flow proceeds to processing block 810, which completes the transaction by supplying the requested data portion to the private storage of processor i. Otherwise, control flow proceeds to processing block 804. In processing block 804, if the requested data portion has a status of EC, then control flow proceeds to processing block 810, which supplies the requested data portion to the private storage of processor i. Otherwise, control flow proceeds to processing block 805.

**[0058]** In processing block 805, if the requested data portion has a status of ED, then control flow proceeds to processing block 806, which requests an updated data portion from the private storage of the processor that did not initiate the data request. Following the receipt of an updated copy, a new status encoding, of M for instance, may be reassigned to the data portion. Alternatively the data portion may be written back to central storage or main memory and a new status of EC or S may be reassigned to the data portion. Control flow then proceeds to processing block 810, which supplies the requested data portion to the private storage of processor i.

**[0059]** If the data portion does not have a status of ED in processing block 805, then control flow proceeds instead to processing block 807. In processing block 807, if the data portion has a status encoding of I, then control flow proceeds to processing block 808, which requests an external data portion

from central storage or main memory. Following the receipt of the data portion from central storage or main memory, control flow proceeds to processing block 810, which completes the transaction by supplying the requested data portion to the private storage of processor i.

**[0060]** It will be appreciated that tradeoffs and modifications may be made by those skilled in the art without departing from the principles of the present invention within the scope of the accompanying claims. For instance, a tradeoff may be made between the amount of information encoded by a presence encoding and the average number of unnecessary invalidation requests transmitted. A similar tradeoff may be made between the presence encodings and the average number of backward data requests transmitted to private storage.

**[0061]** Figure 9a illustrates an alternative embodiment of a coherent storage hierarchy including private storage 910, private storage 920, private storage 940, private storage 980, and shared storage 990 that supports resolving sharing ambiguities. Control 991 includes a status portion and a presence portion for each corresponding data portion stored by shared storage 990. The presence portion of control 991 holds a presence encoding, which indicates either the two high-order private storage units 906 or the two low-order private storage units 905. If an invalidation request must be transmitted, control 991 transmits requests to both of the low-order private storage units 905 when the least significant bit of a presence encoding is set and to both of the high-order private storage units 906 when the most significant bit of a presence encoding

is set. Thus half as much storage is needed to store presence encodings and potentially twice as many invalidation requests are transmitted. Potentially five thirds the average number of backward data requests are also transmitted to provide coherence when a requested data portion has a status of ED.

**[0062]** For example, data portion 911, data portion 913, data portion 915 and data portion 918 have statuses of ED; so if one of these data portions is requested by a private storage unit identified with the same presence encoding, then the data portion has already been provided to the other private storage unit identified with that presence encoding in response to a data request of a type that indicates a need to modify the requested data portion. On the other hand, if the requesting private storage unit is not indicated by the presence encoding, then a data request for an updated copy will be transmitted to both of the private storage units indicated by the presence encoding. Since ED is an exclusive status, only one will respond.

**[0063]** Status portions corresponding to data portion 912 and data portion 919 hold status encodings of S and their corresponding presence portions hold presence encodings of 11 indicating that these data portions have been provided to at least two of the private storage units in response to data requests that did not need to modify the requested data portion. If invalidation requests need to be transmitted for either of these data portions, then all four of the private storage units will receive a transmission when, it is possible that only two or three of the private storage units actually need to invalidate their data portions.

[0064] Data portion 914 has a status encoding of M and a presence encoding of 10 indicating that the previously modified data portion 914 has been provided to at least one of the high-order private storage units 906 in response to a data request of a type that does not indicate a need to modify the requested data portion. Data portion 916 has a status encoding of I and a presence encoding of 00 indicating that data portion 916 has been invalidated in shared storage and any private storage which may have previously held a copy.

[0065] Data portion 917 and data portion 971 have status encodings of EC and presence encodings of 01 indicating that the corresponding data portions have been provided to at least one of the low-order private storage units 905 in response to a data request of a type that does not indicate a need to modify the requested data portion.

[0066] Further tradeoffs and modifications may be made in alternative embodiments by one skilled in the art. For instance, it will be appreciated that the amount of information encoded by a three-bit status encoding permits up to four exclusive-dirty statuses to be encoded. For a four-processor system, each exclusive-dirty status encoding could identify the private storage unit holding the most up-to-date copy of a data portion.

[0067] Figure 9b illustrates another alternative embodiment of a coherent storage hierarchy including private storage 910, private storage 920, private storage 940, private storage 980, and shared storage 990. Control 992 includes a status portion for each corresponding data portion stored by shared

storage 990. The status portion of control 992 holds a status encoding, which indicates an exclusive-dirty status for each private storage unit, ED1 for private storage 910, ED2 for private storage 920, ED4 for private storage 940, and ED8 for private storage 980. If an invalidation request must be transmitted, control 992 transmits requests to all four of the private storage units. Thus, no unnecessary backward data requests are transmitted when a requested data portion has an exclusive-dirty status. No additional storage is needed to store presence encodings but potentially four times as many invalidation requests are transmitted.

**[0068]** For example, a status portion corresponding to data portion 921 holds a status encoding of ED1 indicating that the corresponding data portion has been provided to the private storage 910 in response to a data request of a type that indicates a need to modify the requested data portion. Data portion 923 has a status encoding of ED4 indicating that the corresponding data portion has been provided to the private storage 940 in response to a data request that indicates a need to modify the requested data portion. Data portion 925 has a status of ED8 indicating that the corresponding data portion has been provided to the private storage 980 in response to a data request that indicates a need to modify the requested data portion. Additionally, data portion 928 has a status of ED2 indicating that the corresponding data portion has been provided to the private storage 920 in response to a data request of a type that indicates a need to modify the requested data portion.

[0069] Data portion 922 and data portion 929 have status encodings of S indicating that the corresponding data portion has been provided to one or more of the private storage units in response to a data request that does not need to modify the requested data portion. If invalidation requests need to be transmitted for either of the corresponding data portions, then all four of the private storage units will receive a transmission when, in fact, it is possible that only one, two or three of the private storage units actually need to invalidate the requested data portion.

[0070] Data portion 924 has a status of M indicating that the previously modified data portion 924 may be provided to a private storage unit in response to a data request that does not indicate a need to modify the requested data portion. Data portion 926 has a status encoding of I indicating that data portion 926 has been invalidated in shared storage and any private storage which may have previously held a copy. Data portion 927 and data portion 972 have status encodings of EC indicating that the these data portion have each been provided to a private storage unit in response to a data request that did not indicate a need to modify the requested data portion.

[0071] Figure 9c illustrates another alternative embodiment of a coherent storage hierarchy. Control 993 includes a status portion and a presence portion for each corresponding data portion stored by shared storage 990. The status portion of control 993 holds a status encoding, which includes an exclusive-dirty status for each private storage unit, ED1 for private storage 910, ED2 for private storage 920, ED4 for private storage 940, and ED8 for private

storage 980. The presence portion of control 993 holds a presence encoding, which includes a high-order bit for high-order storage units 906 including private storage 940 or private storage 980, or a low-order bit for low-order storage units 905 including private storage 910 or private storage 920. If an invalidation request must be transmitted, control 993 transmits requests to two of the private storage units for each presence encoding bit that is set. Thus, no unnecessary backwards data requests are transmitted when a requested data portion has an exclusive-dirty status and half as much additional storage is needed to store presence encodings but potentially twice as many invalidation requests are transmitted.

**[0072]** For example data portion 931 has a status encoding of ED1 and a presence encoding of 01 indicating that data portion 931 has been provided to low-order private storage 910 in response to a data request of a type that indicates a need to modify the requested data portion. Data portion 933 has a status encoding of ED4 and a presence encoding of 10 indicating that data portion 933 has been provided to high-order private storage 940 in response to a data request that indicated a need to modify the requested data portion. Data portion 935 has a status encoding of ED8 and a presence encoding of 10 indicating that data portion 935 has been provided to high-order private storage 980 in response to a data request that indicated a need to modify the requested data portion. Data portion 938 has a status encoding of ED2 and a presence encoding of 01 indicating that data portion 938 has been provided to low-order private storage 920 in response to a data request of a type that

indicates a need to modify the requested data portion. If invalidation requests or backwards data requests need to be transmitted for any of these data portions, then only the appropriate private storage unit will receive a transmission.

**[0073]** Data portion 932 and data portion 939 have status encodings of S and presence encodings of 11 indicating that these data portions have been provided to at least two of the private storage units in response to data requests that did not indicate a need to modify the requested data portion. If invalidation requests need to be transmitted for either of these data portions, then all four of the private storage units will receive a transmission when, in fact, it is possible that only two or three of the private storage units may actually need to invalidate their data portions.

**[0074]** Data portion 934 has a status encoding of M and a presence encoding of 10 indicating that the previously modified data portion 934 has been provided to at least one high-order private storage unit in response to a data request that did not indicate a need to modify the requested data portion. If invalidation requests need to be transmitted for data portion 934, then the two high-order private storage units 906 will receive a transmission when, it is possible that only one of the two private storage units may actually need to invalidate their data portions.

**[0075]** Data portion 936 has a status encoding of I and a presence encoding of 00 indicating that data portion 936 has been invalidated in shared storage and any private storage which may have previously held a copy.

[0076] Data portion 937 and data portion 973 have status encodings of EC and presence encodings of 01 indicating that these data portions have both been provided to at least one of the low-order private storage units 905 in response to data requests that did not indicate a need to modify the requested data portion. If invalidation requests need to be transmitted for either of these data portions, then two of the private storage units will receive a transmission when, it is possible that only one of the two private storage units may actually need to invalidate their data portions.

[0077] It will be appreciated that elements such as protocols and communication methods can be modified in arrangement and detail by those skilled in the art without departing from the principles of the present invention within the scope of the accompanying claims. For instance, distributed storage systems and distributed coherence protocols may enjoy the benefits of reduced backward inquiries and locally shared control consolidation.

[0078] Figure 9d illustrates another alternative embodiment of a coherent storage hierarchy in a networked distributed system including private storage 901, private storage 902, private storage 904, private storage 908, and shared storage 990 that supports resolving sharing ambiguities. Control 994 includes a status portion and a presence portion for each corresponding data portion stored by shared storage 990. The status portion of control 994 holds a status encoding, which includes one exclusive-dirty status. The presence portion of control 994 holds a presence encoding, which comprises a doubly linked SCI list.

[0079] For example a status portion corresponding to data portion 951 holds a status encoding of ED and a corresponding presence portion holds a presence encoding for list51 indicating that data portion 951 has been provided to the private storage at the head of list51 in response to a data request of a type that indicates a need to modify the requested data portion. Data portion 953 has a status encoding of ED and a presence encoding for list53 indicating that data portion 953 has been provided to the private storage at the head of list53 in response to a data request that indicated a need to modify the requested data portion. Data portion 955 has a status encoding of ED and a presence encoding for list55 indicating that data portion 955 has been provided to the private storage at the head of list55 in response to a data request that indicated a need to modify the requested data portion. Data portion 958 has a status encoding of ED and a presence encoding for list58 indicating that data portion 958 has been provided to the private storage at the head of list58 in response to a data request that indicated a need to modify the requested data portion.

[0080] Data portion 952 has a status encoding of S and a presence encoding for list52 indicating that data portion 952 has been provided to the private storage units of list52 in response to data requests of a type that does not indicate a need to modify the requested data portion. Data portion 959 has a status encoding of S and a presence encoding for list59 indicating that data portion 959 has been provided to the private storage units of list59 in response

to data requests of a type that does not indicate a need to modify the requested data portion.

**[0081]** Data portion 954 has a status encoding of M and a presence encoding for list54 indicating that previously modified data portion 954 has been provided the private storage units of list54 in response to a data request of a type that does not indicate a need to modify the requested data portion.

**[0082]** Data portion 956 has a status encoding of I indicating that data portion 956 has been invalidated in shared storage and a presence encoding for list56, which may or may not be significant. It will be appreciated that in a distributed system some delay may be expected to invalidate private storage copies and that it may be convenient to hold some information in presence encoding list56, but in general a status encoding of I indicates that no valid data or presence encodings are available for the corresponding storage location.

**[0083]** Data portion 957 has a status encoding of EC and a presence encoding for list57 indicating that data portion 957 has been provided exclusively to the private storage at the head of list57 in response to a data request of a type that does not indicate a need to modify the requested data portion. Data portion 974 has a status encoding of EC and a presence encoding for list74 indicating that data portion 974 has been provided exclusively to the private storage at the head of list74 in response to a data request of a type that does not indicate a need to modify the requested data portion.

[0084] Figure 10 illustrates an embodiment of a computing system 1000 including a coherent storage hierarchy comprising private storage 1011, 1012, 1016 and 1017 using an M-S-I coherence protocol, shared storage 1010 and 1015 using an M-ED-EC-S-I coherence protocol that supports resolving sharing ambiguities and shared storage 1004 using bus snooping on a shared bus 1003 and an M-E-S-I coherence protocol. Multiprocessing systems 1001 and 1002 may be on a single or on multiple printed circuit boards. Alternatively, multiprocessing systems 1001 and 1002 may be on a multiple-processor-core silicon die, or in a multi-chip module.

[0085] Figure 11 illustrates another embodiment of a computing system 1100 including a coherent storage hierarchy comprising private storage 1111, 1112, 1116 and 1117 using an M-S-I coherence protocol, shared storage 1110 and 1115 using an M-ED-EC-S-I coherence protocol that supports resolving sharing ambiguities and distributed shared storage 1104 using an SCI coherence protocol. Multiprocessing systems 1101 and 1102 may be arranged so as to appear to be single processor systems to each other and to shared storage 1104 when communication via communication network 1103.

[0086] Figure 12 illustrates another embodiment of a computing system 1200 including a coherent storage hierarchy comprising private storage 1211, 1212, 1216 and 1217 using a distributed SCI coherence protocol, distributed shared storage 1204 also using a distributed SCI coherence protocol, and distributed shared storage 1210 and 1215 using an M-ED-EC-S-I coherence protocol that supports exclusive locally isolated ownership while resolving

sharing ambiguities. Multiprocessing systems 1201 and 1202 may be arranged in a manner similar to that shown in Figure 9d. Multiprocessing systems 1201 and 1202 may also be arranged so as to appear to be single processor systems to each other and to shared storage 1204 when communication via local area network (LAN) 1203. Multiprocessing systems 1201 and 1202 may provide different types of access to shared storage 1204. For instance, multiprocessing system 1201 may provide a more public access over a wide area network (WAN), while multiprocessing system 1202 may provide a more secure access over a virtual private network (VPN).

**[0087]** It will be appreciated that the methods and apparatuses herein disclosed may be used in multiple user multiprocessing systems or in single user multiprocessing systems or in multiple core processor. Figure 13a illustrates an embodiment of multiple core processor 1351 including a coherent storage hierarchy 1301 comprising private storage 1310, private storage 1320 and shared storage 1390 having storage control 1391 that supports resolving sharing ambiguities. Private storage 1310 and private storage 1320 correspond to processor core 1311 and processor core 1312 respectively. Processor core 1311 and private storage 1310 may be closely integrated into a combined processing core 1315. Similarly processor core 1312 and private storage 1320 may be closely integrated into a combined processing core 1325. It will be appreciated that multiple core processor 1351 may comprise a single die or may comprise multiple dies and that processing core 1315 may be similar or dissimilar to processing core 1325. It will also be appreciated

multiple core processor 1351 may further comprise bus control circuitry or other communication circuitry, processor cores in addition to processor cores 1311 and 1312 and private storage in addition to private storage 1310 and 1320. Figure 13b further illustrates an embodiment of computing system 1302 including a coherent storage hierarchy comprising private storage 1310, 1320, ...1340 and shared storage 1390 having storage control 1391 that supports resolving sharing ambiguities. Private storage 1310, 1320, ... 1340 correspond to processors 1321, 1322, ... 1340 respectively. Computing system 1302 may comprise a personal computer including but not limited to central processing 1352, graphics storage, other cache storage and local storage; system bus(ses), local bus(ses) and bridge(s); peripheral systems, disk and input/output systems, network systems and storage systems.

**[0088]** The above description is intended to illustrate preferred embodiments of the present invention. From the discussion above it should also be apparent that the invention can be modified in arrangement and detail by those skilled in the art without departing from the principles of the present invention within the scope of the accompanying claims.